厦门新市民核验-X

2022年12月08日

更新日志

时间	说明
20221111	**接口上架**
20221111	**接口下架**
20221112	**接口上架**

1. 接口信息

1.1厦门新市民核验-X

URL: /government/economic/352

REQUEST TYPE : get,post

REQUEST PARAM :

Param name	Param type	Required	Desc
key	string	是	您购买的API的key值
idcard	string	是	身份证号码
name	string	是	姓名
authorized	string	是	已经取得客户授权，传1，否则传0。未取得客户授权不返回数据。
pushurl	string	是	回调的URL地址（长度不超过255个字符）。系统将以POST回调。数据格式：{

RESPONSE PARAM :

Param name	Param type	Desc
code	string	请求code码
message	string	code码说明
data	string	接口返回数据体
seqNo	string	调用唯一标识（如有接口问题，请提供此值）

RESPONSE PARAM data :

Param name	Param type	Desc
msg	string	
data	string	
citizentype	string	新市民类别，0-非新市民；新市民1有户籍类新市民2无户籍有居住证 3无户籍无居住证
expiredate	string	预留
score	string	正常返回数字；如果返回

SUCCESS RESPONSE :

```
{
  "code":"0",
  "message":"成功",
  "data":
  {
    "citizentype": "0",
    "expiredate": "",
    "score": "632"
  },
  "seqNo":"F72EFZZL2211012218"
}
```

ERROR CODE :

Code value	Desc

-1	传入参数无效
-10	过期的请求
-11	无效签名
-12	没有数据
-13	网络连接超时
-2	Key无效
-3	AccessToken无效
-4	临时拒绝访问
-5	拒绝访问
-6	无客户授权
-7	系统忙
-8	内部错误
-9	重复的请求
0	成功
10000	成功

2. 全系统错误码

Code value	Desc
SYSTEM_900	IP 不合法
SYSTEM_999	接口处理异常
SYSTEM_000	key 参数不能为空
SYSTEM_001	找不到这个 key
SYSTEM_002	调用次数已用完
SYSTEM_003	用户该接口状态为不可用
SYSTEM_004	接口信息不存在
SYSTEM_005	你没有认证信息
SYSTEM_008	当前接口只允许“企业认证”通过的账户进行调用，请在厦门大数据产品服务平台官网个人中心进行企业认证后再进行调用，谢谢！
SYSTEM_009	必须认证审核通过才可以使用
SYSTEM_011	接口缺少参数
SYSTEM_012	没有 ip 访问权限
SYSTEM_013	接口模板不存在
SYSTEM_014	接口模板没开启
SYSTEM_015	该接口已下架
SYSTEM_017	模板配置的平台参数与请求的参数不一致
SYSTEM_019	调用第三方协议配置错误
SYSTEM_020	调用第三方产生异常
SYSTEM_022	调用第三方返回的数据格式错误
SYSTEM_025	你没有购买此接口
SYSTEM_026	用户信息不存在
SYSTEM_027	请求第三方地址超时，请稍后再试
SYSTEM_028	请求第三方地址被拒绝，请稍后再试
SYSTEM_029	返回示例错误
SYSTEM_034	签名不合法
SYSTEM_035	请求参数加密有误
SYSTEM_036	验签失败
SYSTEM_037	timestamp 不能为空

SYSTEM_038	请求繁忙，请稍候再试
SYSTEM_039	请在个人中心接口设置加密状态
SYSTEM_040	timestamp 不合法
SYSTEM_041	timestamp 过期
SYSTEM_042	身份证手机号等不符合规则
SYSTEM_043	当前您的接口覆盖范围不能满足本次数据验证，请升级 API 接口套餐获取更丰富的服务

3. 接口对接示例代码

3.1 sample code

```
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.config.RequestConfig;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.util.EntityUtils;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class HttpUtil {
    public static void main(String[] args) {
        //接口地址，以下 url 为案例，以网页展示的实际 url 为准
        String url = "https://api.xmcic.cn:51888/trade/user/1985";
        //请求参数
        Map<String, Object> params = new HashMap<>();
        //输入厦门大数据产品服务平台提供的 key
        params.put("key", "");
        String result = null;
        try {
            result = post(url, params);
        } catch (Exception e) {
            e.printStackTrace();
        }
        System.out.println("result:\n" + result);
    }
    public static String post(String url, Map<String, Object> params) throws Exception
{
        ArrayList<NameValuePair> pairs = covertParams2NVPS(params);
        return PostHttpRequest(url, pairs);
    }
    public static String PostHttpRequest(String Url, List<NameValuePair> params) throws
```

```
Exception {
        CloseableHttpClient client = HttpClients.createDefault();
        //超时时间
        RequestConfig requestConfig = RequestConfig.custom()
                .setSocketTimeout(300000)
                .setConnectTimeout(300000)
                .build();
        String result = null;
        try {
            HttpPost request = new HttpPost(Url);
            request.setConfig(requestConfig);
            request.setEntity(new UrlEncodedFormEntity(params, "UTF-8"));
            HttpResponse respones = client.execute(request);
            if (respones.getStatusLine().getStatusCode() == 200) {
                result = EntityUtils.toString(respones.getEntity(), "UTF-8");
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            client.close();
        }
        return result;
    }
    private static ArrayList<NameValuePair> covertParams2NVPS(Map<String, Object>
params) {
        ArrayList<NameValuePair> pairs = new ArrayList<>();
        if (params == null || params.size() == 0) {
            return pairs;
        }
        for (Map.Entry<String, Object> param : params.entrySet()) {
            Object value = param.getValue();
            if (value instanceof String[]) {
                String[] values = (String[]) value;
                for (String v : values) {
                    pairs.add(new BasicNameValuePair(param.getKey(), v));
                }
            } else {
                if (value instanceof Integer) {
                    value = Integer.toString((Integer) value);
                } else if (value instanceof Long) {
                    value = Long.toString((Long) value);
                }
                pairs.add(new BasicNameValuePair(param.getKey(), (String) value));
            }
        }
        return pairs;
    }
}
```

3.2 获取 key

登陆 https://dsj.xmcic.cn:8089/,首页 > 个人中心 > 我的接口，如下图，举例 红色箭头指向处为对应接口名称的 key

4. 加密对接说明

我们的加密方式使用 MD5 进行 sign 签名验证，以此来杜绝信息篡改的发生，同时针对入参和出参内容进行对称加密（base64 编码）。以下具体讲解对接方式。

对接步骤如下：

4.1 设置加密

第一步：登陆 https://dsj.xmcic.cn:8089/,首页 > 个人中心 > 基础服务 > 我的账户 点击“获取密钥”，如下图，举例 secretKey:lgJKiiakuplMXy4s （注意！这是秘钥与 key 不同，且本文秘钥为案例，请通过下图获取正式密钥）

第二步：首页 > 个人中心 > 基础服务 > 我的 API 选择某个接口操作栏中的“数据服务”按钮，弹框中选择“加密对接”

4.2 技术对接

接口的入参和出参参数名无需使用 base64 编码，只是针对入参值和出参值使用密码加解密。案例如下：

Url: https://api.xmcic.cn:51888/communication/personal/1896
RequestWay: Post
SecretKey : lgJKiiakuplMXy4s
Request Param:

名称	类型	必填	说明
key	string	是	您购买的 API 的 key 值
name	string	是	姓名
idcard	string	是	身份证号

e.g
key=您购买的 API 的 key 值
name=谢天浩&idcard=342623198610025939&timestamp=1505352152882

开始处理如下：

第一步、BASE64AES 加密入参 ： 备注：key 字段不参与加密

name=zmTbaQJ6cV5VMgapQOlC4w==&idcard=a+AwqXMTUOdOqaiawhMCpVoJkSNFzZxpCT1HiVbTIHM=&timestamp=lD5VSE5xkjgH88Sjl3FQSw==

第二步、sign 令牌获取 ：

规则为入参按照 ACS 码排序,结果如下：

idcard=a+AwqXMTUOdOqaiawhMCpVoJkSNFzZxpCT1HiVbTIHM=&name=ZmTbaQJ6cV5VMgapQOlC4w==&timestamp=lD5VSE5xkjgH88Sjl3FQSw==

备注：key 字段不参与签名

sign=new BASE64Encoder().encode(md5(入参按照 ACS 码排序结果)) 获得入参

sign=//WpqjHlIQ2lKJtb3oajYA==

第三步、开始发送请求 ：

url：https://api.xmcic.cn:51888/communication/personal/1896

requestWay:post

Request:

key=您购买的 API 的 key 值

name=ZmTbaQJ6cV5VMgapQOlC4w==&idcard=a+AwqXMTUOdOqaiawhMCpVoJkSNFzZxpCT1HiVbTIHM=&timestamp=lD5VSE5xkjgH88Sjl3FQSw==

sign=//WpqjHlIQ2lKJtb3oajYA==

Response:

```
{
    "code": "10000",
    "message": "成功",
    "data": "L58pwLVQ7L8ZarUVOgpCLA==",
    "seqNo": "C03SX59Z1709071444"
}
```

data 值秘钥进行 BASE64AES 解密，解密结果如下：{"state": "1"}

4.3 java 代码示例

Maven pom Repositories:

```
<dependency>
 <groupId>com.cdp.product</groupId>
 <artifactId>cdp-common-security</artifactId>
 <version>3.5.0</version>
</dependency>
```

BASE64AES 加密工具类和方法：

com.cdp.product.security.encode.CdpEncryptUtil.aesEncrypt(明文,秘钥)

BASE64AES 解密工具类和方法：

com.cdp.product.security.decode.CdpDecryptUtil.aesDecrypt(密文,秘钥) Sign

签名工具类和方法：CdpSignUtil.sign(Map<String, String> param) 加密解密代

码示例如下：

cdp-common-security 下载地址：

https://www.xmcic.cn/static/cdp-common-security-3.5.0.zip

```
package com.cdp.product.security.test;

import com.cdp.product.security.decode.CdpDecryptUtil;
import com.cdp.product.security.encode.CdpEncryptUtil;
import com.cdp.product.security.exception.DecryptFailureException;
import com.cdp.product.security.exception.EncryptFailureException;
import com.cdp.product.security.exception.SignFailureException;
import com.cdp.product.security.sign.CdpSignUtil;
```

```
import java.util.HashMap;
import java.util.Map;

/**
 * 测试加解密以及签名
 */
public class CommonTest {
    public static void main(String[] argc) throws EncryptFailureException,
SignFailureException, DecryptFailureException {
        //秘钥
        String secretKey = "lgJKiiakuplMXy4s";
        //入参集合 针对入参 value 值进行加密
        Map<String, String> param = new HashMap<>();
        param.put("name", CdpEncryptUtil.aesEncrypt("谢天浩", secretKey));
        param.put("idcard", CdpEncryptUtil.aesEncrypt("342623198610025939", secretKey));
        param.put("timestamp", CdpEncryptUtil.aesEncrypt(System.currentTimeMillis() +
"", secretKey));
        //获取 sign 签名
        String sign = CdpSignUtil.sign(param);
        //请求参数
        Map<String, Object> params = new HashMap<>();
        params.putAll(param);
        //输入厦门大数据产品服务平台提供的 key
        params.put("key", key);
        //输入 sign 签名
        params.put("sign", sign);
        //发起请求
        result = post(url, params);
        //返回各种加解密签名结果
        System.out.println(String.format("入参集合:\n%s", param));
        System.out.println(String.format("获取 sign 值:\n%s", sign));
        System.out.print(String.format("返回结果解密:\n%s",
        CdpDecryptUtil.aesDecrypt("81hwJ3Fzj4De9fNJccustQ==", secretKey)));
    }
}
```

结果示例:

入参集合:

{timestamp=JD5VSE5xkjgH88Sjl3FQSw==,name=ZmTbaQJ6cV5VMgapQ0lC4w
==, idcard=a+AwqXMTUOdOqaiawhMCpVoJkSNFzZxpCT1HiVbTIHM=}

获取 sign 值:

//WpqjHlIQ2lKJtb3oajYA==

返回结果解密:

{"state": "1"}